AD 697 847

CRT-AIDED SEMI-AUTOMATED MATHEMATICS

James H. Bennett, et al

Applied Logic Corporation
Princeton, New Jersey

October 1969

Qualified requestors may obtain additional copies from the Defense Documentation Center. All others should apply to the Clearinghouse for Federal Scientific and Technical Information.

AFCRL-69-0464

CRT-AIDED SEMI-AUTOMATED MATHEMATICS

by

James H. Bennett   Francis C. Oglesby
James R. Guard   Mark L. Bayern, Jr.

APPLIED LOGIC CORPORATION
ONE PALMER SQUARE
PRINCETON, NEW JERSEY 08540

Contract F-19628-69-C-0064

Project No. 5632
Task No. 563205
Work Unit No. 56320501

FINAL REPORT

Period Covered: June 1, 1968 through August 26, 1969

October 1969

Contract Monitor: Timothy Hart, Data Sciences Laboratory

Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce, for sale to the general public.

Prepared

for

AIR FORCE CAMBRIDGE RESEARCH LABORATORIES
OFFICE OF AEROSPACE RESEARCH
UNITED STATES AIR FORCE
BEDFORD, MASSACHUSETTS 01730

ABSTRACT

This report describes the status of the sixth in a series of six experiments in semi-automated mathematics. This effort extended from June 1, 1968 through August 26, 1969. These experiments culminated in large complex computer programs which allow a mathematician to prove mathematical theorems on a man-machine basis. SAM VI, the sixth program, uses a cathode ray tube as the principal interface between the mathematician and a high speed digital computer. An elaborate language and logical capability has been implemented in SAM VI. These include I/O languages for expressing mathematical statements in a form suitable for both the mathematician and the machine to recognize and handle with ease and convenience, a language for expressing and handling sorts and range of symbols, and an autologic algorithm and matching routine. The latter contain the capability for handling, automatically, logic with equality.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

The authors wish to acknowledge the encouragement given during every phase of the work by their technical monitor, Timothy P. Hart.

PERSONNEL

The authors wish to express appreciation for the whole-hearted efforts on the part of the following Applied Logic personnel:

William B. Easton

Thomas H. Mott, Jr.

Aaron B. Kronenberg

INTRODUCTION

Since 1960, a group of mathematicians who are now located at Applied Logic Corporation, Princeton, New Jersey, have been studying techniques to use the non-numeric capabilities of a digital computer to enable a mathematician to prove theorems and solve other non-numeric tasks. This endeavor has culminated in six large computer programs, called SAM I, II, III, IV, V, VI. The last, SAM VI, is only partially implemented. The logical framework of SAM is an ω-order calculus with typed and many-sorted variables, equality, and λ-notation. SAM is currently running on Applied Logic's time-sharing AL/COM computer network (PDP-10) using a CRT, light pen, and Teletype for interaction. The basic philosophy behind this work has been that theorem proving and many other symbol manupulative problems were too difficult for realistic solutions by existing computers running without human intervention. It seems to us that methods for allowing a mathematician to interact with a non-numeric computer calculation might produce results which could not be achieved by either party alone.

The development of the SAM programs is guided by the principle of adding the new feature, or improving the old feature, which would make SAM most useful (and tolerable) to the mathematician. For example, in spite of the convenience of using an existing higher level language, we chose to write SAM in assembly language, thereby getting as much speed as possible from the automatic portions of SAM. Even such sympathetic SAM users as ourselves find long delays in proving a lemma intolerable. For the same reason, when concepts arise which are encountered frequently, special purpose software is written rather than attempting to have a lesser amount of more general software. For example, equality is not treated as just some binary relation with three axioms, but rather the rules of equality are very strongly built into the algorithms of SAM. In addition, if SAM proves that a binary operation is commutative and/or associative, the algorithms in SAM VI can automatically consider various combinations of permuting or associating the arguments of such operations. Similarly, a dynamic structure of many-sorted type variables can be specified by the mathematician using SAM.

The development of an experimental tool to study AUTOLOGIC, the automatic logic portion of SAM, has been one of the major objectives of the SAM V and VI programs. During the course of this work various experimental control and debugging routines have been added to SAM--for example, routines for changing weighting functions, creating libraries of formulas, setting various program parameters such as the number of levels of multiple-digressions, and carrying out various housekeeping chores. Because of the experimental or debugging nature of these routines, their control had been largely through the PDP-6/10's debugging language, DDT. Many of the functions which these routines perform, however, have turned out to be important for the operation of SAM and must be made available to the non-programming user. We have thus implemented a comprehensive control package which allows persons to run SAM without understanding SAM's inner workings. This "front end" also permits remote users without a CRT to operate SAM in a fairly natural manner.

During the past year, support for the SAM project has been at a much lower level than in previous years, and we have thus concentrated our efforts upon the control-input/output interface between SAM and the user. The preliminary version of the front end which was reported on in [2], has been completely rewritten during the past year. As in the preliminary version, SAM's current front end is implemented in SNOBOL, thereby maintaining the flexibility which SNOBOL coding gives for building in new features in the future with a minimum of bother.

Section I describes in detail the features available in SAM's current front end and how these features along with the CRT and the light pen are used to operate SAM. Some familiarity with references [1] and [2] is assumed in section I.

Section II indicates briefly some areas in which future investigation would be desirable.

SECTION I

OPERATOR'S GUIDE FOR SAM

The current SAM program should be run in at least 16K (23K if DDT and the symbol table are loaded). Any additional core will be added to the words available for TROLL-lists. Upon starting the program, a call is made to a general dynamic space allocator to set up SNOBOL string space, TROLL-list space, TTY buffers, etc., resulting in about 2K of core being added to the program. The front end signals that it is ready to accept a command by typing a "*".

The first commands given should SET SCOPE and DRUMIO to appropriate values (see the SET command below) and then the INITIALIZE command should be given. (The front end will recognize any unique initial segment of a command verb. Thus, for example, it is sufficient to type INI.)

An initial list of reductions (LR) can be set up in various ways. The most usual way is to INSERT PSDs from the TTY. (PSDs are described on page 6 of [1]. PSDs are often called clauses in the literature and the disjuncts of a PSD are called literals.) For example, the axioms for a group and the closure assumption for a subset H could be inserted as follows (underlined symbols typed by SAM):

```
* INSERT
P (P(X,Y),Z)=P(X,P(Y,Z))
$
L

P(E,X)=X
$
L

P(F(X),X)=E
$
L

NOT(H(X))
NOT(H(Y))
H(P(X,Y))
$
A

*
```

The "$" sign serves as the end of PSD marker. The "L" signals the input routine to return for another insert after adding the PSD to the LR as an axiom. The PSD is also added as an axiom if "A" is given, but control then returns to the front end. If "C" had been typed instead of "A" in the above example, than the PSDs representing the logical negation of the PSD would have been inserted, namely the three PSDs H(X2Ø), H(Y2Ø) and NOT (P(X2Ø,Y2Ø)) where X2Ø and Y2Ø are new individual constants.

If a list of PSDs has been kept from a previous run, then LR can be set up by using the GET command (for binary-mode files obtained with the SAVE command) or by using the READ command (for ASCII files obtained with the WRITE command). For example, to read, using unit number 3, into the LR the PSDs in the ASCII file FILNAM, the following commands would be given:

```
* OLDFILE 3 FILNAM

* READ 3

*
```

The SET and DECLARE commands are used to change the settings of various program parameters or to specify various I/0 options. For example:

```
* SET WEIGHT LENGTH + 3 DISJUNCTS
```

will change SAM's weighting function from the default setting of simply the length of a PSD to the length plus 3 times the number of disjuncts in the PSD. As another example, the group in the example above can be made Abelian either by INSERTing the commutative axiom, P(X,Y)=P(Y,X), or by giving the command DECLARE COMMUT P (the latter usually being preferable since the commutativity of P is then treated internally in the matching algorithms). The RESET command can be used to reinitialize parameters and declarations, and their present state can be output by using the EXAMINE, TYPE, or PRINT commands. In addition, current settings and declarations can be saved in an ASCII disc-file by the command WRITE with argument PARAMETERS. The resulting file can then be READ back in at some later date, thus re-establishing the current settings and declarations.

Having inserted an initial LR and made appropriate changes to parameters, AUTOLOGIC is then turned loose with the START command to generate consequences of these initial PSDs. If the CRT is available, the user is able to watch the progress of SAM on the scope. The display shows a section of the proof and a set of buttons. By using the light pen on the display buttons (or alternatively by using the equivalent TTY commands), the user can control the display and the action of SAM. He can cause any section of the proof to be displayed, have the proof "roll" UP or DOWN, TRACK on the end of the LR, i.e., display the lowest PSDs on the list, etc. By setting SPEAK to TTY or LPT, the user can follow SAM's progress even without a CRT.

The proof procedure can be halted either by obtaining a light-pen hit on the HALT button or by interrupting from the TTY by typing any character (e.g., space). The user can then INSERT or DELETE PSDs, TYPE out the current state of various lists or parameters, make changes in such things as the weighting function or the I/O options, request a proof HISTORY, WRITE or SAVE lists of PSDs for later use, etc. The START command can then be used to continue the proof.

In the remainder of this section we give an alphabetic listing of the commands currently available in SAM along with a brief description of each. Unless otherwise indicated, command is given from TTY.

<u>CHIS</u>. Closes current history file (if any). See IHIS.

<u>DDT</u>. Transfers control to DDT if loaded. (Return from DDT is START$G.)

<u>DECLARE</u> NAME LIST. NAME can be COMMUT, ASSOC, INFIX, WSYMBOL, MATH, OR GREEK.

In the case of the first four of these, LIST should be a list of symbols being declared. Thus, for example,

DECLARE COMMUT A H2 S1

will declare the (binary) symbols A, H2, and S1 to be commutative to the internal matching algorithms of AUTOLOGIC. INFIX F will cause the binary symbol F to be output in the infix (central) position (e.g. $X \wedge Y$ if F is $\wedge$).

WSYMBOL declares the symbols in LIST to e special weighting symbols used by the weighting function SDENSITY (see WEIGHT under the SET command).

If NAME is MATH or GREEK, then LIST should be a list of ordered pairs. For example,

DECLARE GREEK (G2 A) (D K)

will make any output (on the scope) of G2 be as an alpha and D as a kappa. For a complete description of special output symbols, see the Appendix.

The command UNDECLARE NAME LIST is similar except it removes declarations.

DELETE NUMBER. Deletes PSD with indicated number from LR. Complains if can't find the number in LR. Command can be given with light pen or TTY. As a TTY command, "DELETE CONTRA" can be given. The effect of this is to remove all consequences of an INSERTed logical negation from the LR and to REMOVE the LE.

DOWN. Similar to UP.

EXAMINE NAME. Type out value(s) from NAME (which can be as in RESET).

EXIT. This command generates an exit call to the system, closes all open files, and results in the user being put in monitor command mode and the teletype responding with

```
EXIT
↑C
.
```

GET. Program requests file name. The present LR is junked and the named file is read into the LR along with any proof analysis of its PSDs. Assumes the extension LR if none given. File must be in binary mode (same as used by drum I/O and the HISTORY and SAVE commands), and DRUMIO must be ON.

HALT. Light pen command only. Stops a running program. Control stays with the light pen.

HDELETE NUMBER. Available only if a history file is currently active (see IHIS). Similar to DELETE except in addition to deleting the PSD with the indicated number from LR, all descendants of this PSD are deleted from the LR and from the LE (including DRUMLE files).

HISTORY NUMBER. If no history file is currently active (see IHIS), then no action is taken. Otherwise the proof of the indicated PSD from LR is displayed on the scope if the scope is active; if no scope, then output is on the TTY. Complains if no number is given or if the number cannot be found in LR. (List containing the history is junked only upon starting SAM and hence can be printed on the line printer or written in a disc file by using the PRINT or WRITE commands). Command can be given with light pen or TTY.

IHIS. Program requests name for history file and then initializes program to write history file in FILE.HIS while SAM is running. Always uses extension HIS. DRUMIO must be ON.

NEWFILE UNIT NAME.EXT. Opens new ASCII disc file called NAME.EXT on unit number UNIT. If NAME.EXT already exists, it will be overwritten.

OLDFILE UNIT NAME.EXT. Opens existing ASCII disc file NAME.EXT on unit number UNIT. Write pointer is set for appending onto file.

POINTER NUMBER. Positions LOW pointer in LR to point at the PSD with the indicated number. Complains if no number given or if can't find the number in LR. Command can be given with light pen or TTY.

PRINT LR
SORT
HISTORY
PARAMETERS

Same as TYPE except output is to line printer.

READ UNIT. Same as INSERT except that PSDs in the ASCII file open on unit number UNIT are inserted into LR. Analyses (if present in the disc file) are cleared. An additional argument PARAMETERS can be given. In this case the file open on UNIT should be a parameter file created by the WRITE command. Reading of such a file will return parameter settings and declarations to their state at the time the WRITE command was given.

RELEASE UNIT. Closes file open on unit number UNIT and releases the unit.

REMOVE LR
LE
SORT

The indicated list is removed and its cells are returned to available space. Complains if no list is given. (LE includes deleting DRUMLE files).

RESET NAME. Reset NAME to its default value (see SET command). NAME can be any of the possible NAMES from the SET or DECLARE commands (e.g. DIGRESS, SCALE, ASSOC, INFIX). If no name is given, then all values (except SCOPE and DRUMIO) are reset (however, lists - e.g. LR - are not reset).

REWEIGHT. Recomputes weights of all PSDs on LR and LE (including DRUMLE files) and reorders LE according to the new weights.

SAVE. Program requests file name and then outputs the LR in binary mode (same as used by drum I/O) into the file named. The extension LR is added if no extension is given. (Note: in binary mode, the GET command will re-establish the proof analyses of PSDs.) DRUMIO must be ON.

SCOPE. Transfers control from TTY to scope light-pen, if scope exists.

SET NAME VALUE.

Sets appropriate parameters of SAM as follows:

NAME	Possible Values	Default Value
SCOPE	OFF, ON	OFF
DRUMIO	OFF, ON	OFF
DIGRESS	OFF, ON, 2 thru 6	OFF
SPEAK	OFF, TTY, LPT	OFF
WEIGHT	(see below)	(see below)
LWINDOW	non-negative integers	Ø
UWINDOW	Positive integers	1ØØ
SCALE	any integer	Ø

Setting SCOPE ON lets SAM know that the CRT is being used and hence the CRT will be initialized, formulas will be displayed while SAM is running, and light-pen control will be available. DRUMIO should be set ON if the user wishes to have the drum input/output routines available for extending the list of expansions to external storage as explained on p. 15 of [2]. DRUMIO must also be ON if the HISTORY, LIBRARY, SAVE, or GET commands are being used. The INITIALIZE command must be given after a change in either the SCOPE or DRUMIO setting.

Giving DIGRESS the value ON results in the normal digression process being included in the AUTOLOGIC algorithms. N-step multiple digression (see pp. 2-3 of [2]) is turned on by giving DIGRESS the value N, $2 \leq N \leq 6$.

If SPEAK has value TTY, then a history of PSDs appearing on the list of reductions is typed out as SAM is running. Similarly if SPEAK has value LPT, this history is printed on the line printer. (If the SPEAK feature is used, interrupting from the TTY must be done by executing a "control C" and then a REENTER for the SPEAK feature).

The VALUE of WEIGHT should be a linear weighting function of the form:

$$N_1 \ FUN_1 + \ldots + N_K \ FUN_K,$$

where each N_i is an integer (positive or negative) and FUN_i is one of the following functions of PSDs: LENGTH (=total number of symbols in PSD), DISJUNCTS (=number of disjuncts in PSD), DEPTH (=number of levels PSD is from the axioms in the current proof), VDENSITY (=LENGTH divided by the number of occurrences of variables in PSD), SDENSITY (=LENGTH divided by the number of occurrences of symbols in PSD declared as WSYMBOLS), or CONTRA (=Ø if PSD is a consequence of the logical negation of something we are trying to prove, =1 otherwise). The current weighting function is stored internally in SAM and used along with the value of SCALE (see below) to compute the weight of PSDs as they are generated. PSDs are melded into the list of expansions according to this weight, small weights being placed on top, and hence the weight of a PSD serves as a criterion of importance. (Since the weighting routine is a separate module, experimentation with more sophisticated weighting functions is easily accomplished by the user familiar with DDT.) The default VALUE of WEIGHT is LENGTH. The command REWEIGHT should usually be given after making a change in the setting of WEIGHT.

LWINDOW and UWINDOW are used to throw away expansions (or digressions) whose weights are outside some desired range. Thus PSDs with weight less than LWINDOW or greater than UWINDOW will not be generated.

In computing the weight of a PSD, the number obtained from the weighting function is multiplied by two raised to the value of SCALE and then truncated to be a positive number less than $1ØØ_8$.

START. Starts algorithms of AUTOLOGIC. If START command is given after program is interrupted, program will be continued at the point it left off if that is possible. If program cannot be continued (e.g., PSDs deleted), then AUTOLOGIC is started at the top of its main loop. The LOW pointer in LR will remain the same if the PSD pointed at still exists. Otherwise LOW is set at the top or the bottom of LR according as LE is empty or not. Command can be given with light pen or TTY.

TOP. Similar to UP except the topmost PSDs of LR are displayed.

TRACK. Similar to UP except the bottom PSDs on LR are displayed followed (after a marker) by the current top PSD of the LE.

TTY. Light pen command only. Transfers control to TTY.

TYPE LR
SORT
HISTORY
PARAMETERS

If argument is a list, it is output to the TTY in PSD form. If argument is PARAMETERS, the current values of all locations which can be modified by using the SET or DECLARE commands are output.

UNDECLARE. See DECLARE

UP. Light pen command (also available to TTY), active only if SCOPE is ON. Moves display window up six PSDs.

WRITE UNIT LR
SORT
HISTORY
PARAMETERS

Same as TYPE except that output is to the ASCII file open on unit number UNIT and each PSD (if argument is a list) is followed by a "$" (end of PSD sign) in order that the file may be read later. If no file is open on UNIT, then a created file, FILEN.DAT where N=UNIT, is used.

DANGER. Available on CRT only. Interesting effect; worth trying once.

SECTION II

THOUGHTS ON FUTURE DEVELOPMENT

SAM's success with such first-order systems as group theory, linear algebra, and lattice theory affirms the basic value of the semi-automated approach to theorem-proving. However, much remains to be accomplished if SAM is to become an efficient, practical mathematical tool, operating at what we consider to be the most natural level --i.e., natural deduction in a many sorted calculus of order ω. In this section we indicate briefly some areas in which further work is needed.

The matching routines of AUTOLOGIC need to be extended to handle functions of higher type. A more general sort structure than the one currently implemented in SAM VI should be added, and new sorted variables and new theorems about the containment relations between sorts should be generated automatically as well as be specified by the user. The AUTOLOGIC algorithm should be extended to allow tree-structured proofs (as opposed to just linearly structured proofs). Of particular importance is the need for building special arithmetic routines into SAM for the efficient handling of counting arguments and arbitrarily large, but finite, mathematical entities. Mathematicians rarely work on 2X2 matrix theory, but rather nXn matrix theory; a basic theorem of finite group theory states that the order of a subgroup divides the order of the group; and so on.

Input/output languages must be correspondingly extended to express quantifiers, sort specifications, proof-trees, etc. The current front end permits SAM to be operated by the non-programming user, but it clearly will require further refinement to remain responsive to the needs of the user. Thought should be given to the possibility of allowing the user the flexibility of modifying or specifying (at least in part) the syntax handling in SAM by means of a SNOBOL or a SNOBOL/LISP interpretive front end.

A number of interesting problems are to be found in such an effort. As one example, how does one display mathematical calculations in a way that the mathematician can monitor SAM and yet allow SAM to perform its calculations effectively. For certain types of mathematics, this problem has been partially solved by the SAM programs.

In Figure 1 we see some consequences of axioms for group theory much as they would appear on the screen of the CRT display tube. In this case the internal representations of the formulas are quite closely connected with the external form displayed and therefore it is a reasonably simple matter to display in a perspicuous manner calculations being done by the automatic portion of SAM. In this simple algebraic setting the notations used by mathematicians are reasonably close to a full formal notation needed by SAM.

However, other types of mathematical shorthand or notation which are commonly used are really quite distant from a formalization of the mathematics being displayed. For example, Figure 2 is a mockup of a possible computation in SAM VI. 00001 and 00002 represent axioms concerning "less than" and the fact that the function F is positive. 00003 represents the theorem to be proven. Notice the form of 00003. Two different inductions are disguised in this common looking statement. 00004 through 00013 represent internal steps which are automatically generated in order to prove the proposed theorem from the two axioms. Even in this simple proof, 00011, seems somewhat remote from 00001, 00002, and 00003. In a more difficult theorem, where the assistance of the mathematician may be needed, considerable problems arise in terms of displaying internal portions of proofs in some recognizable notation.

Turning our attention to counting arguments and arbitrarily large, but finite, mathematical entities, the mock-up in Figure 2 gives a flavor of some of the development needed. Let us examine this example in more detail.

00004 is a restatement of 00003 which must be automatically generated. From internal information, SAM must know that N, N_1 and the values of F range over integers, and that $<$ is a central-binary predicate whose arguments are real numbers. Since the integers are a subsort of the reals (and SAM must either know or prove this), 00004 seems syntactically an intermediate step toward proving 00003. Steps 00006, 00007, and 00005 form a subordinate proof which gives the definition of a recursive function G as hypothesis and a restatement of 00004 as the conclusion to prove. The remaining steps of the proof are trivial.

```
00001   0<N → N1<N1+N                                          AXM
00002   0 < F(N)                                               AXM
00005    |  |G(0) = F(0)                                       REC 00004
00007    |  |G(N3+1) = G(N3)+F(N3+1)                           REC 00004
00010    |  |  |- G(N22) < G(N22+1)                            SKL 00005
00011    |  |  |- G(N22) < G(N22)+F(N22+1)               00007 RED 00010
00012    |  |  |- 0 < F(N22+1)                           00001 EXP 00011
00013    |  |  |FAL                                      00002 RED 00012
00005    |  |G(N2) < G(N2+1)                                   REC 00004
00004    |F(0)+... +F(N2) < F(0)+... +F(N2+1)                  REC 00003
00003   F(0) < F(0)+F(1) <.... < F(0)+... +F(N1)               THM
```

00001	$(\alpha \circ \beta) \circ \gamma = \alpha \circ (\beta \circ \gamma)$	AXM
00002	NOT $(\alpha \circ \gamma = \alpha \circ \beta)$ $\gamma = \beta$	AXM
00003	$^{-1}(\alpha) \circ \alpha = \wedge$	AXM
00004	$\wedge \circ \alpha = \alpha$	AXM
00006	$^{-1}(\alpha) \circ (\alpha \circ \epsilon) = \epsilon$	00001 RED OF 00005 (02)
00011	$^{-1}(\wedge) = \wedge$	00003 EXP OF 00010 (05)
00015	$^{-1}(^{-1}(\epsilon)) = \epsilon$	00012 RED OF 00014 (05)
00016	$\epsilon \circ \wedge = \epsilon$	00015 RED OF 00012 (07)
00017	$\epsilon \circ {}^{-1}(\epsilon) = \wedge$	00003 EXP OF 00015 (07)
00021	$\epsilon \circ (^{-1}(\epsilon) \circ \alpha) = \alpha$	00001 RED OF 00020 (11)

APPENDIX

SPECIAL DISPLAY SYMBOLS

In order to allow expression of formulas with symbols which are most natural or familiar, there is the capability of displaying symbols as lower case Greek letters and common mathematical symbols on the CRT display. See DECLARE in Section I for instructions on how to use this facility. The tables below show the symbols currently available and their "transliterations", the ASCII characters used to designate them. Additional sets of symbols can be added by simple program modifications. (Hard copy with special symbols can also be made on an incremental plotter, see [1] pp. 23-26.)

Greek

Grk	Tr	Grk	Tr
α	A	ν	N
β	B	ξ	C
γ	G	o	O
δ	D	π	P
ϵ	E	ρ	R
ζ	Z	σ	S
η	H	τ	T
θ	Q	υ	U
ι	I	ϕ	F
κ	K	χ	X
λ	L	ψ	V
μ	M	ω	W

Math

Mth	Tr	Mth	Tr	Mth	Tr
$\wedge$	K	$\cup$	J	$\bullet$	*
$\vee$	A	$\cap$	M	$\dashv$	,
$\sim$	N	$\subseteq$	P	$\oplus$	+
$\rightarrow$	C	$\subsetneq$	Q	$\otimes$	X
$\equiv$	E	$\supseteq$	R	$\succ$	G
$\forall$	&	$\supsetneq$	S	$\approx$	I
$\exists$	$\$$	ε	#	$\prec$	L
$\leq$	<	$\{$	(	$\vert$	!
$\geq$	>	$\}$	)	$\Vert$	/
$\neq$	=	‘‘	T	$\triangle$	@
		‘	U		
		’	V		
		’’	W		

BIBLIOGRAPHY

1. "CRT-Aided Semi-Automated Mathematics" by J. H. Bennett, W. B. Easton, J. R. Guard, and L. G. Settle. Final Report No. AFCRL-67-0167. January 1967. (Contract No. AF 19(628)-3250).

2. "CRT-Aided Semi-Automated Mathematics" by J. H. Bennett, J. R. Guard, R. Haydock, F. C. Oglesby, W. L. Paschke, and L. G. Settle. Final Report dated July 1968. (Contract No. AF F-19628-67-C-0100).

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*
Applied Logic Corporation, One Palmer Sq.
Princeton, New Jersey 08540

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
CRT-AIDED SEMI-AUTOMATED MATHEMATICS

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
Scientific Final. 1 June 1968 - 26 August 1969 approved 30 Oct. 1969

5. AUTHOR(S) *(Last name, first name, initial)*
James H. Bennett
James R. Guard
Francis C. Oglesby
Mark L. Bayern, Jr.

6. REPORT DATE
October 1969

7a. TOTAL NO. OF PAGES
27

7b. NO. OF REFS
2

8a. CONTRACT OR GRANT NO.
F19628-69-C-0064

b. PROJECT, Task, Work Unit Nos.
5632-05-01

c. DoD Element: 61102F

d. DoD Subelement: 681305

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
AFCRL-69-0464

10. AVAILABILITY/LIMITATION NOTICES
1. Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce, for sale to the general public.

11. SUPPLEMENTARY NOTES
TECH, OTHER

12. SPONSORING MILITARY ACTIVITY
Air Force Cambridge Research Laboratories (CRB)
L. G. Hanscom Field
Bedford, Massachusetts 01730

13. ABSTRACT
This report describes the status of the sixth in a series of six experiments in semi-automated mathematics. This effort extended from June 1, 1968 ~~through~~ August 26, 1969. These experiments culminated in large complex computer programs that ~~which~~ allow a mathematician to prove mathematical theorems on a man-machine basis. SAM VI, the sixth program, uses a cathode ray tube as the principal interface between the mathematician and a high speed digital computer. An elaborate language and logical capability has been implemented in SAM VI. These include I/0 languages for expressing mathematical statements in a form suitable for both the mathematician and the machine to recognize and handle with ease and convenience, a language for expressing and handling sorts and range of symbols, and auto-logic algorithm and matching routine. The latter contain the capability for handling, automatically, logic with equality.

DD FORM 1473 1 JAN 64

Unclassified
Security Classification

Unclassified
Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Man-machine mathematics Semi-automated mathematics Mathematical displays on CRT						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year, or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through ________________."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through ________________."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through ________________."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS), (S), (C), or (U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.

Unclassified
Security Classification